REMARKS

[0002] Applicant respectfully requests reconsideration and allowance of all of the claims of the application. The status of the claims is as follows:

- Claims 1-9, 11-15, 17-22, 24-34, 36-43, 45 and 51 were pending
- Claims 6, 12, 19, 26, and 38 are amended herein
- Claims are 1-9, 11-15, 17-22, 24-34, 36-43, 45 and 51 are currently pending

[0003] Amendments to the claims consist of corrections of minor informalities or deletion of elements from the claims. No new matter is introduced.

Entry After Final Office Action

[0004] Entry of Applicant's amendments and remarks after Final is appropriate under 37 C.F.R. §1.116 because no new issue is raised. The amendments either correct minor informalities or remove already examined portions of the claims thereby placing the rejected claims in better form for consideration on appeal.

Claim Objections

[0005] Claim 6 stands objected to as allegedly for a minor informality. Applicant amends the claims as suggested by the Examiner to obviate the objection.

Cited Documents

[0006] The following documents have been applied to reject one or more claims of the Application:

- Koved: Koved, et al., "IBM Research Report Access Rights Analysis for Java", IBM, October 31, 2001, cover page and pp 1-13
- Wong:: Wong, et al. , "Secure Programming with .NET", retrieved on 8-4-2009 at <<http://www.securityfocus.com/ infocus/1645>>, Security Focus, November 26, 2002, pp 1-5

Claims 1-9, 11-15, 17-22, 24-34, 36-43, 45 and 51 Are Non-Obvious Over Koved in view of Wong

[0007] Claims 1-9, 11-15, 17-22, 24-34, 36-43, 45 and 51 stand rejected under 35 U.S.C. §103(a) as allegedly being obvious over Koved in view of Wong. Applicant respectfully traverses the rejection.

Independent Claim 1

[0008] Applicant submits that the Office has not made a *prima facie* showing that independent claim 1 is obvious in view of the combination of Koved and Wong. Applicant notes that it is only after the USPTO makes a demonstration of unpatentability that the burden shifts to the applicant to rebut that showing. *In re Oetiker*, 977 F.2d 1443, 1445 (Fed. Cir. 1992) ("[T]he examiner bears the initial burden, on review of the prior art or on any other ground, of presenting a *prima facie* case of unpatentability. If that burden is met, the burden of coming forward with evidence or argument shifts to the applicant.").

[0009] Applicant submits that the combination of Koved and Wong does not teach or suggest at least the following features of this claim (with emphasis added):

A method implemented on a computing device having instructions stored on a computer-readable storage media and executable by a processor, to estimate security requirements needed to execute a managed code for a developer prior to an actual execution of the managed code, comprising:

simulating the execution of all calls from an assembly to another assembly for all execution paths of one or more assemblies in the managed code, wherein the assembly comprises one or more files versioned and deployed as a unit, wherein the managed code is a managed shared library or an executable, wherein all managed code is contained within the one or more assemblies, wherein ***the execution of each assembly is statically simulated without actually running a corresponding managed code to simulate all possible calls and corresponding flow of argument data***;

finding a set of required permissions for each execution path by one or more simulated stack walks that each include a plurality of the assemblies, wherein each call in each execution path has a corresponding permissions set, wherein each assembly has one or more execution paths representing a different data and a control flow, and wherein ***the simulated stack walk comprises:***

entering an execution path corresponding to a static simulation of execution of the assembly;

entering a public entry point of a method in the assembly;

gathering a permission set for the method in the assembly;

determining whether the method in the assembly calls another method in the assembly or in an another assembly;

gathering a permission set for the another method called by the method in the assembly; and

creating a union of the gathered permission sets; and

deriving the security requirements for execution paths corresponding to the one or more assemblies by using the union of the gathered permission sets across the execution paths corresponding to the one or more assemblies, wherein the union estimates the security requirements that will be triggered against the one or more assemblies during the actual execution of the one or more assemblies and whether a security exception will be triggered during the actual execution.

[0010] Claim 1 recites in part, "the execution of each assembly is statically simulated without actually running a corresponding managed code to simulate all possible calls and corresponding flow of argument data." As the grounds of rejection the Office cites Koved, page 1, Abstract and page 2, column 2 as allegedly disclosing this element. (Office Action, page 3.) Applicant respectfully disagrees.

[0011] Applicant respectfully notes: "[W]hat a reference teaches is a question of fact." *Amazon.com, Inc. v. Barnesandnoble.com, Inc.*, 239 F.3d 1343, 1358 (Fed. Cir. 2001) (referencing *In re Beattie,* 974 F.2d 1309, 1311 (Fed. Cir. 1992)). See also *McGinley v. Franklin Sports*, 262 F.3d 1339, 1350 (Fed. Cir. 2001). The Office has made a finding of fact that the abstract of Koved discloses "test cases... cover all paths". (Office Action, page 3.) However, the abstract of Koved actually recites that "test cases usually **do not** cover all paths through the code, so failures can occur in deployed systems" (emphasis added). Applicant respectfully submits that the Office by mischaracterizing the Koved reference has failed to make a finding of fact sufficient to support the conclusion of unpatentability. The abstract of Koved discusses exactly the opposite of Applicant's claimed simulation of all possible calls and corresponding flows or argument data. Accordingly, this grounds of rejection is traversed.

[0012] Moreover, Applicant has searched and failed to find any disclosure in page 2, column 2 of Koved of "the execution of each assembly is statically simulated without actually running a corresponding managed code to simulate all possible calls and corresponding flow of argument data," as recited in Applicant's claim 1. The Office provides no explanation as to how the cited portion of Koved corresponds to the actual claim language. Since the Office has not cited the reference with specificity and provided no reasoning for its rejections, Applicant is forced to make assumptions and guesses as to the Office's specific reasoning. Therefore, Applicant submits that it has been denied its right to adequately and effectively respond to the Office's rejections. Applicant submits that the Office has not articulated the reasons for its decision-making here. Furthermore, according to 37 CFR §1.113 and MPEP §706.07, Applicant respectfully submits that no clear issues has been developed between the Applicant and the Office for each pending claim so that such issues would be ready for appeal.

[0013] The Office has provided no evidence or arguments that Wong compensates for these deficiencies in Koved. Accordingly, the rejection of Applicant's claim 1 is traversed.

[0014] Claim 1 further recites in part, that "*the simulated stack walk comprises: entering an execution path corresponding to a static simulation of execution of the assembly; entering a public entry point of a method in the assembly; gathering a permission set for the method in the assembly; determining whether the method in the assembly calls another method in the assembly or in an another assembly; gathering a permission set for the another method called by the method in the assembly; and creating a union of the gathered permission sets*." The Office acknowledges that Koved

does not disclose these features. (Office Action, page 4). As the grounds of rejection the Office cites Wong, page 2, "Stack Walking" as allegedly disclosing a static simulation of execution of the assembly. (Office Action, page 5.) Applicant respectfully disagrees. The "Stack Walking" section of Wong describes:

> When are the permissions generated by the code-access security module checked? The permissions are checked during a process known as a stack walk. A stack walk operates as follows. When a new method is called, a new activation record containing the return address, the parameters passed to the method, and any local variables is placed on the stack. This record is popped off the stack on returning from the method. As a result, the stack grows and shrinks during the course of program execution. Before granting access to a method, a protected resource may demand a stack walk which, as the name implies, entails walking through all the records in the call chain and determining if they have appropriate access rights for the requested resource. This procedure ensures a higher level of safety than checking only the permission set of the immediate caller.

[0015] Even assuming arguendo that Wong discusses stack walking, Applicant can find nothing in the cited portion of Wong that discloses a static stack walk. In contrast, the cited portion of Wong is directed to an operation performed "when a new method is called" not a static simulation of execution as recited in Applicant's claim.

[0016] As the grounds of rejection the Office further cites Wong, page 3, "Requesting minimum required permissions for execution" and "Securing sensitive information" as allegedly disclosing gathering a permission set for the method in the assembly. (Office Action, page 5). Applicant respectfully disagrees. The cited sections of Wong describe:

> 1. Requesting minimum required permissions for execution

The .NET application programmer can explicitly specify the minimum level of required permissions for the code to execute. In this case, if the code encounters security policies that do not grant it permission to execute, the code will not run. This facility prevents ugly exception details from being thrown at the user midway through execution and can be achieved by appropriate use of the SecurityAction.RequestMinimum method. The programmer should also specify to the system that no additional permissions (more than those required for the legitimate execution of the code) be granted to the code. This prevents widening of security holes elsewhere in the code. This is analogous to using a UNIX account with lowest possible privileges to perform a particular task. This can be achieved by setting Unrestricted=false in the permission set.

2. Securing sensitive information

Applications that handle sensitive data need to prevent exposure of that data to malicious code. While code access security might stop malicious code from accessing resources, such code could still read values of fields or properties that might contain sensitive information. In order to keep data secure in memory it should be stored as private or internal variables, thus limiting its scope to the same assembly. The programmer must, however, be aware of the fact that the data can still be accessed by highly trusted code under reflection, serialization and debugging.

Data can also be secured as “protected”, limiting its access to that particular class and its derivatives. However, the programmer must take necessary steps to ensure that all derived classes implement similar protection. Such controlled inheritance can be achieved using the InheritanceDemand feature.

[0017] The Office has provided no reasoning for its rejections; Applicant is forced to make assumptions and guesses as to which portions of the cited reference the Office is mapping to elements of Applicant's claims. Applicant intuits that the Office may be equating "[a] programmer can explicitly specify the minimum level of required permissions for the code to execute" from Wong with "*gathering a permission set for the method in the assembly*," of Applicant's claim 1. However, Wong discusses an affirmative step that must be performed by a programmer (i.e., specification) with an operation performed by a computer-implemented method (i.e., gathering).

[0018] The Office may also be interpreting "SecurityAction.RequestMinimum method" from Wong as mapping to "*gathering a permission set for the method in the assembly*." However, the name of this method is misleading. In .NET the SecurityAction.RequestMinimum value indicates that an assembly requires the permission set to operate. Without the permission set there is no point in loading the assembly. (*See* Löwy, Juval, *Programming .NET Components*, O'Reilly & Associates, Inc., 2003, p. 380). In other words, the SecurityAction.RequestMinimum method states what is required (i.e., minimum permissions) to run an assembly. In contract Applicant's claim recites "*gathering a permission set*" which is not disclosed in Wong.

[0019] For all the above reason, the Office Action does not establish a *prima facia* case that combination of Koved and Wong teaches or suggests all of the elements and features of this claim. Accordingly, Applicant respectfully requests that the rejection of this claim be withdrawn.

Independent Claims 12, 19, 26, and 38

[0020] Independent claims 12, 19, 26, and 38 are similar to independent claim 1, and thus, non-obvious over the cited documents of record for at least the same reasons as independent claim 1. Consequently, the combination of Koved and Wong does not teach or suggest all of the elements and features of these claims. Accordingly, Applicant respectfully requests that the rejection of these claims be withdrawn.

Dependent Claims 2-9, 11, 13-15, 17-18, 20-22, 24-25, 27-34, 36-37, 39-43, 25, and 51

[0021] Claims 2-9, 11, 13-15, 17-18, 20-22, 24-25, 27-34, 36-37, 39-43, 25, and 51 ultimately depend from one of independent claims 1, 12, 19, 26, or 38. As discussed above, claims 1, 12, 19, 26, and 38 are patentable over the cited documents of record. Therefore, claims 2-9, 11, 13-15, 17-18, 20-22, 24-25, 27-34, 36-37, 39-43, 25, and 51 are also patentable over the cited documents of record for at least their dependency from a patentable base claim. These claims may also be patentable for the additional features that each recites.

Conclusion

[0022] Applicant submits that all pending claims are in condition for allowance. Applicant respectfully requests reconsideration and prompt issuance of the application. If any issues remain that prevent issuance of this application, the Examiner is urged to contact the undersigned representative for the Applicant before issuing a subsequent Action.

Respectfully Submitted,

Lee & Hayes, PLLC
Representative for Applicant

/kaseychristie40559/ Dated: October 5, 2009

Benjamin A. Keim
(benjamink@leehayes.com; 509-944-4748)
Registration No. 59,217

Reviewer/Supervisor: Kasey Christie
(kasey@leehayes.com; 509-944-4732)
Registration No. 40,559